US005752032A

United States Patent [19]

Keller et al.

[11] Patent Number: 5,752,032

[45] Date of Patent: May 12, 1998

[54] **ADAPTIVE DEVICE DRIVER USING CONTROLLER HARDWARE SUB-ELEMENT IDENTIFIER**

[75] Inventors: **James A. Keller**, Santa Clara; **Kevin J. Flory**, Patterson, both of Calif.

[73] Assignee: **Diamond Multimedia Systems, Inc.**, San Jose, Calif.

[21] Appl. No.: **560,801**

[22] Filed: **Nov. 21, 1995**

[51] **Int. Cl.**[6] **G06F 13/10**

[52] **U.S. Cl.** **395/681**; 395/651; 395/828; 395/284

[58] **Field of Search** 395/828, 830, 395/831, 651, 653, 681, 682, 284

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,649,479	3/1987	Advani et al.	364/300
4,755,478	7/1988	Abernathey et al.	437/41
4,855,936	8/1989	Casey et al.	364/521
4,885,259	12/1989	Osinski et al.	437/41
4,975,829	12/1990	Clarey et al.	364/200
5,175,855	12/1992	Putnam et al.	395/700
5,265,252	11/1993	Rawson, III et al.	395/700
5,305,461	4/1994	Feigenbaum et al.	395/775
5,337,412	8/1994	Baker et al.	395/275
5,339,432	8/1994	Crick	395/651
5,352,631	10/1994	Sitaram et al.	437/200
5,412,798	5/1995	Garney	395/500
5,459,867	10/1995	Adams et al.	395/681
5,459,869	10/1995	Spilo	395/700
5,465,364	11/1995	Lathrop et al.	395/681
5,491,813	2/1996	Bondy et al.	395/500
5,530,858	6/1996	Stanley et al.	395/650
5,555,401	9/1996	Allen et al.	395/500
5,564,011	10/1996	Yammine et al.	395/182.13
5,581,766	12/1996	Spurlock	395/652
5,586,324	12/1996	Sato et al.	395/652
5,590,314	12/1996	Ueno et al.	395/681
5,603,014	2/1997	Woodring et al.	395/500

Primary Examiner—Glenn A. Auve

Attorney, Agent, or Firm—Fliesler Dubb Meyer & Lovejoy

[57] **ABSTRACT**

A device driver architecture that couples an operating system to a computer interface of a controller device that includes a plurality of functional sub-elements. The device driver includes a plurality of operating system interface objects each presenting an operating system interface (OSI) to the operating system, a plurality of computer interface objects each providing for the generation of programming values to be applied to the computer interface to establish the operating mode of a respective predetermined sub-element of the controller device, and a device driver library of processing routines callable by each of the plurality of operating system interface objects to process data and generate calls to the plurality of computer interface objects in predetermined combinations. The device driver library enables the selection of an execution contexts within which to define the generation and application of the programming values to the computer interface. The state of the hardware interface is virtualized and maintained in discrete contexts, allowing for application specific, dynamic alteration of the state of the hardware interface through essentially context switching private to the device driver in response to selected operating system events.

10 Claims, 8 Drawing Sheets